**Doubts:**

* How to define frontmatter and metadata.
* How to distribute and monitor tasks.
* How to manage sites for different documents. (within Metamation domain)
* How to handle videos in .md files.
* How to collect feedback on document quality.

**Input Channel:**

* Whenever a new documentation request raises, the same is notified to the Tech Doc team using Fogbugz cases.
  + Fogbugz is a project management tool used at Metamation.
* Create a new case in Fogbugz to initiate a document creation process.
  + Fogbugz [link](https://metamation.fogbugz.com/login?dest=%2ff%2fsearch%2f%3fsColumns%3dCategory-Favorite-Case-TitleComment-Project-Status-OpenedBy-Priority-AssignedTo-RemainingTime%26sSearchFor%3dpraxis%26sSorts%3dRemainingTime-Case.descending).
* To create a case, get necessary approval and access to use Fogbugz.
  + Contact [sanjiv@metamation.com](mailto:sanjiv@metamation.com)

**Technical Documentation:**

* Technical Documentation process at Metamation can be bifurcated into three stages. They are:
  + Initial/Preparation stage.
  + Version Control System
    - Git
    - GitHub
  + Editorial and Publishing System
    - Hugo – Static site generator
    - Render – Website hosting

**Initial/Preparation stage:**

1. Based on the Tech Doc request, identify the document type that needs to be created.
2. Liase with SMEs and stakeholders to collect all available inputs.
3. Have a Q&A session with SME to clarify any doubts.
4. Once you have everything ready, move to next stage.

**Version Control System:**

* **VCS** is a software designed to help team collaborate and work in parallel.
* **Git** is distributed version control software. Version control is a way to save changes over time without overwriting previous versions. Being distributed means that every developer working with a Git repository has a copy of that entire repository - every commit, every branch, every file. Git is used locally in individual systems.
* **GitHub** is a cloud server used for team collaboration and task monitoring.

**GitHub:**

1. Create a GitHub account.
   * + Onetime process.
2. Create a standalone repository.
   * + A standalone repository must be created for each document.
3. Define standard rules for repo and branches.

**Git:**

1. Create input and output folders and organize the input files in compliance with folder structure.
2. Initiate an empty Git repository by *git init*.
3. Clone the repository. (git clone)
4. Establish remote connection of the repository with .git folder and sync all the available files to GitHub cloud.
   * Type the following in the cmd prompt and press enter for each code of line.
     + git remote add origin <repo link>
     + git push -u origin main
     + git push –set –upstream origin main
5. To sync, use Git Push:
   * + Git commit
     + Git add . (To add all files to the staging area)

**Editorial and Publishing System:**

**Hugo:**

Hugo is a general-purpose website framework. Technically speaking, Hugo is a static site generator. Unlike systems that dynamically build a page with each visitor request, Hugo builds pages when you create or update your content. Since websites are viewed far more often than they are edited, Hugo is designed to provide an optimal viewing experience for your website’s end users and an ideal writing experience for website authors.

Getting started: <https://gohugo.io/getting-started/quick-start>

**Steps:**

1. Create a new site.
   * *hugo new site <sitename>*
   * Download the hugo theme ‘Relearn’ and place it inside the themes folder. <https://mcshelby.github.io/hugo-theme-relearn/index.html>
   * Edit the config file to add the theme name.
2. Add content.
   * Create pages: 1. Home page, 2. Chapter & 3. Default page.
     + *hugo new --kind home \_index.md*
     + *hugo new --kind chapter <name>/\_index.md*
     + *hugo new <chapter>/<name>/\_index.md*
   * Introduce markdown syntax. Refer <https://mcshelby.github.io/hugo-theme-relearn/cont/markdown/index.html>
3. Review content and configure the site.
4. Publish the site locally and do a final review.

**Render:**

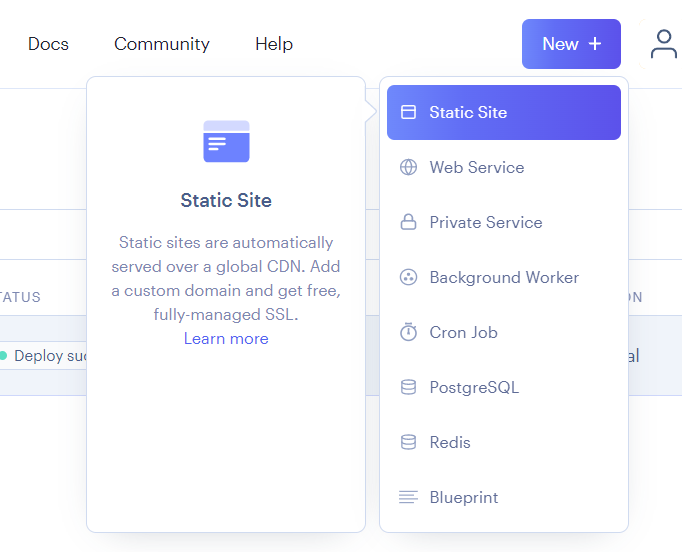
Static Sites

Deploying static sites on Render is incredibly easy. Just link your GitHub or GitLab repo — we build your site every time you push to your repo and serve it over a global CDN with fully managed TLS certificates.

Static sites on Render are free, with no cost at all to you unless you go above 100 GB of bandwidth per month.

**Steps:**

1. Sign into Render using GitHub account. (One time process)
2. Click New and select Static Site.



1. Connect a GitHub repository that you wish to deploy.

Graphical user interface, text, application, email

Description automatically generated

1. Fill in the fields in the following format:
   * Name = <Name of the site>
   * Branch = Main
   * Root directory = <Project folder location>
   * Build command = hugo --gc –minify
   * Publish directory = Public (public folder inside hugo project folder)

Graphical user interface, application, Teams

Description automatically generated

1. Click Create Static site.